Технология и программная поддержка создания Электронных Изданий

С.В.Агошков
ВЦ РАН, Москва, Россия
gosh@eolss.inm.ras.ru

А.В.Котов
ВЦ РАН, Москва, Россия
sanek@ccas.ru

В.А.Серебряков
ВЦ РАН, Москва, Россия
serebr@ccas.ru

К.В. Вигурский
НТЦ "Информрегистр", Москва, Россия
iregistr@mail.sitek.ru

А.Е.Поляков
НТЦ "Информрегистр", Москва, Россия
iregistr@mail.sitek.ru

А.А. Штольберг
НТЦ "Информрегистр", Москва, Россия
shtolberg@mail.ru

Введение

Электронные библиотеки являются специализированным классом ИС. Их назначением является хранение, обработка и представление пользователю содержимого информационных ресурсов различных типов. В свою очередь, среди электронных библиотек выделяются подкласс систем, основными задачами которого является

- Хранение,
- Манипулирование,
- Представление

электронных ресурсов, являющихся аналогами бумажных печатных изданий. В статье такие электронные библиотеки называются “Электронные издания” (ЭИ). Организация информации в электронном издании наследуется из своего “неэлектронного” прообраза, но, по сравнению с ним, электронные научные издания обладают расширенной функциональностью:

- Возможностью функционирования электронного издания в распределенной среде, в том числе и в среде Internet.
- Возможностью хранения, индексации и представления большого количества (600-700 тыс.) полнотекстовых фрагментов. При этом каждый фрагмент понимается как цельная сущность.
- Возможностью хранения, индексации и представления мета–данных о фрагментах.
- Представлением специфического для цифровых библиотек, но обязательного для выделяемого подкласса, набора пользовательских сервисов включающего:
    - Навигацию по структурам над массивом фрагментов.
    - Просмотр текстов документов.
    - Поиск фрагментов по мета-данным (т.е. атрибутный поиск).
    - Поиск фрагментов по лексике (т.е. лексический поиск).
- Наличием полных текстов в нестандартных алфавитах (диакретические алфавиты, математические символы, и т.д.).
- Возможностью интеграции с глобальными цифровыми библиотеками научного типа, например, ИСИР (http://isir.ccas.ru).

В статье изложено описание замкнутого технологического процесса изготовления таких электронных изданий, обеспечивающего их полный жизненный цикл: начиная от подготовки данных для них и заканчивая поддержкой работающей системы. Технология была создана с использованием компонентно-ориентированного подхода, что позволило выделить необходимые этапы технологического процесса. Она была успешно применена для изготовления ЭНИ «Литературное наследие А.С.Пушкина».

© Вторая Всероссийская научная конференция
ЭЛЕКТРОННЫЕ БИБЛИОТЕКИ:
ПЕРСПЕКТИВНЫЕ МЕТОДЫ И ТЕХНОЛОГИИ,
ЭЛЕКТРОННЫЕ КОЛЛЕКЦИИ
26-28 сентября 2000г., Протвино

1 ПРОГРАММНО-ТЕХНОЛОГИЧЕСКИЙ КОМПЛЕКС ПОДГОТОВКИ ДАННЫХ ДЛЯ ЭЛЕКТРОННЫХ ИЗДАНИЙ.

1.1. Введение

Проблематика подготовки электронных изданий и электронных библиотек разрабатывается в НТЦ "Информрегистр" в течение нескольких лет. Подготовлено и выпущено несколько электронных научных изданий (ЭНИ), отлажен технологический процесс подготовки ЭНИ. В настоящее время ведется работа по созданию системы подготовки электронных изданий, ориентированной на массовое применение и представляющей собой законченный программно-технологический продукт.

Электронные научные издания, сочетая достоинства традиционных печатных изданий с возможностями современных информационных технологий, должны отвечать высоким требованиям качества и характеризоваться следующими свойствами:

- отражать современный уровень науки и эдиционной практики,
- тщательно отбираться и подготавливаться,
- при преобразовании печатных изданий в электронную форму с оптимальной точностью воспроизводить оригинал,
- сопровождаться развитым справочным аппаратом (комментарии, указатели, библиография и т.п.),
- точно идентифицироваться и сопровождаться метаинформацией, соответствующей российским и международным нормам.

Создание электронного научного издания должно начинаться с разработки концепции ЭНИ, которая фиксируется в виде (иерахической) структуры ЭНИ.

Информационная система (оболочка, навигатор) ЭНИ должна обеспечивать пользователю комфортные условия для активной творческой работы с информацией, предоставляя при этом следующие функциональные возможности:

- поддерживать аппарат гипертекстовых и гипермедийных связей;
- осуществлять атрибутный и лексический поиск;
- обладать дружественным пользовательским интерфейсом,
- обеспечивать эффективные средства навигации.

В этом разделе сформулированы основные требования к программно-технологическому комплексу подготовки ЭНИ. Конкретный вариант системы подготовки ЭНИ разрабатывается НТЦ "Информрегистр" совместно с ВЦ РАН. С экспериментальной сетевой версией подготовленного по подобной технологии ЭНИ "Пушкин" можно ознакомиться по адресу http://isir.ccas.ru:81.

1.2. Общие требования

Объект разработки — система подготовки данных для электронных изданий — программно-технологический комплекс (ПТК), обеспечивающий цикл подготовки электронных изданий от исходного материала до комплекта информационных массивов (файлов), готовых к загрузке в результирующую информационную систему.

ПТК совместно с информационной системой (Навигатором) должны составлять продукт, предназначенный для подготовки и выпуска электронных изданий, распространяемых на компактных оптических дисках или через Интернет.

ПТК предназначен для подготовки электронных изданий большого объема: десятки — сотни Мбайт, десятки – сотни тысяч самостоятельно идентифицируемых информационных объектов. ПТК должен быть рассчитан на массовое применение в условиях четко организованного и строго контролируемого технологического процесса, обеспечивающего обработку самих входных данных, генерацию метаданных, а также создание и поддержание структур электронных изданий и входящих в них информационных объектов. Должна быть обеспечена распределенная обработка данных на различных этапах технологического процесса, централизованное управление процессом и контроль данных при переходе от одного технологического этапа к другому.

ПТК должен содержать комплект эксплуатационной документации и описание технологии.

1.3. Входные и выходные данные

ПТК рассчитан на обработку преимущественно текстовой информации, которая может

- включать текст на языках, использующих латинский и кириллический алфавиты,
- иметь сложное форматирование,
- содержать таблицы, формулы, деловую графику (диаграммы, схемы, чертежи и т.п.).

Особым видом текстовой информации, на обработку которой рассчитан ПТК, являются библиографические описания.

Кроме того, ПТК должен обеспечивать обработку изобразительных материалов (рисунков, фотографий и т.п.), которые могут быть как самостоятельными информационными объектами, так и встроенными в другие объекты, в качестве их элементов.

Входные данные могут представляться как на традиционных (бумажных) носителях, так и в электронной форме. В последнем случае данные должны быть представлены в форматах, совместимых с форматом RTF или допускающих конвертирование в RTF стандартными средствами.

Выходные данные должны представлять собой комплект файлов, пригодных для загрузки в информационную систему электронного издания (Навигатор), а также допускающих конвертирование в стандартные форматы типа SGML, XML, HTML, PDF.

1.4. Назначение и основные принципы

1.4.1. ПТК предназначен для специалистов, занятых подготовкой электронных изданий, и должен предоставлять для них:

- удобную среду для обработки информации на всех этапах технологического процесса;
- средства интеллектуального редактирования и контроля обрабатываемой информации;
- средства автоматизации для выполнения рутинных операций при подготовке электронного издания;
- место централизованного хранения массива документов с возможностью распределенной обработки.

1.4.2. При обработке информации в ПТК используется внутренний SGML-подобный язык разметки.

1.4.3. В процессе обработки в ПТК формируются информационные объекты, которые

- могут иметь сложную иерархическую структуру входящих в них элементов,
- могут быть вложенными друг в друга,
- могут быть связанными гипертекстовыми связями (1:1 и 1:n, одно- и двунаправленными).

1.4.4. Все формируемые объекты обладают метаданными (описаниями, атрибутами, именами).

Для формирования метаданных в ПТК используется спецификация Dublin Core Metadata.

Источником описаний для текстовых произведений, представляющих основное содержание информационных объектов, являются их библиографические или эквивалентные им описания.

Источником атрибутов и имен являются описания.

Технологическая схема подготовки метаданных представлена в Приложении 1 (операции 7, 8,9).

1.4.5. Подготавливаемое электронное издание имеет четкую иерархическую структуру, в узлах и листьях (терминальных узлах) которой располагаются информационные объекты.

1.4.6. Структура электронного издания формируется на начальных технологических этапах в процессе разработки концепции ЭИ высококвалифицированными специалистами в соответствующей предметной области (Приложение 1, операция 10). Структура ЭИ фиксируется с помощью средств ПТК и внутреннего языка разметки и определяет глобальную структуру и взаимосвязи выходных данных.

1.4.7. Внутренние структуры информационных объектов формируются и фиксируются в операции 6.

1.5. Схема технологического процесса

ПТК обеспечивает обработку информации по технологической схеме, представленной в Приложении 1, основными операциями которой являются:

1) сканирование и распознавание печатного оригинала (2, 3);

2) проверка (вычитка), корректировка и оформление текстов (4,5);

3) разметка структуры, оформления, гипертекстовых связей (6,16);

4) вторичная вычитка и корректировка текстов после разметки (11);

5) формирование информационных объектов и создание метаданных (7,8,9);

6) формирование структуры электронного издания (10,12).

Технологическая схема может уточняться в процессе разработки ПТК.

1.6. Состав и архитектура ПТК

1.6.1. Общая архитектура

ПТК включает в себя следующие компоненты:

1) стандартную программу распознавания (FineReader);

2) стандартный редактор для обработки RTF-документов (Winword);

3) средства автоматизации разметки;

4) специализированный редактор, настроенный на используемый язык разметки;

5) программу проверки правильности размеченных документов (валидатор);

6) визуализатор размеченных документов (браузер);

7) централизованный репозиторий объектов, включающий средства технологического контроля.

ПТК имеет открытую архитектуру. Все компоненты могут легко настраиваться и перепрограммироваться, а также при необходимости заменяться.

Все компоненты интегрированы в единую среду, которая представляет собой надстройку над Microsoft Office и широко использует его стандартные элементы: редактор (Winword), СУБД (Jet), язык программирования (Visual Basic), а также браузер Internet Explorer.

Ниже перечислены назначение отдельных компонент и требования к ним.

1.6.2. Программа распознавания

Программа распознавания используется для выполнения операций сканирования и распознавания (операции 2–3). На вход ее поступает печатный оригинал, на выходе получается распознанный текст в формате RTF.

1.6.3. Стандартный редактор

Стандартный редактор используется для проверки, корректировки и форматирования распознанного текста в формате RTF (операции 4–5). Основные цели данного этапа обработки таковы:

– выявить и исправить ошибки распознавания;

– воспроизвести текст в виде, максимально близком к печатному оригиналу;

– представить текст в форме, соответствующей определенным технологическим требованиям (см. ниже), необходимым для последующей автоматизированной обработки (разметки).

В качестве стандартного RTF-редактора используется Winword с некоторыми дополнительными надстройками.

1.6.4. Средства автоматизации разметки

Средства авторазметки используются для разметки структуры, оформления и гипертекстовых связей (операции 6 и 16), в автоматическом или полуавтоматическом режиме.

Средства авторазметки позволяют превратить исходный RTF в полностью размеченный документ, используя имеющиеся в тексте формальные признаки (параметры формата, характерные строки символов), в частности:

– автоматически конвертировать элементы оформления RTF (стили, шрифтовое выделение, сноски, таблицы) в соответствующие команды разметки;

– конвертировать стили заголовков RTF в структурные команды разметки;

– автоматизировать разметку основных видов гипертекстовых ссылок (ссылки на комментарии, на указатели и др.).

Средства авторазметки реализуются частично как макросы в среде редактора, частично как внешние программы, вызываемые непосредственно из редактора.

1.6.5. Специализированный редактор

Специализированный редактор используется для массовой обработки размеченных текстов на этапах разметки и проверки (операции 6, 11, 16).

Специализированный редактор предоставляет удобную среду для обработки размеченных текстов, которая обеспечивает следующие возможности:

– цветовое выделение команд разметки с возможностью отключения их показа;

– навигация с учетом логической структуры команд (поиск парной, вышестоящей команды);

– контекстное меню для быстрого ввода команд и атрибутов;

– контекстная помощь для текущей команды или атрибута;

– автоматическая подсветка синтаксических ошибок (незакрытая скобка, кавычка и т.д.);

– автоматическая проверка хотя бы некоторых семантических ошибок по DTD (команда или атрибут не существует, недопустим в данном контексте, и т.д.);

– отображение оформления документа без запуска браузера;

– отображение структуры документа и навигация по его структуре без запуска браузера.

Редактор должен иметь возможность настройки и управления, в частности:

– возможность настройки списков команд и атрибутов, меню, инструментальных панелей, диалогов и других элементов интерфейса через файлы настройки (включая DTD) или макросы;

– включать встроенный язык программирования (типа Visual Basic) или возможность внешнего программирования через технологию ActiveX.

Редактор реализован как надстройка над Winword или как отдельная программа.

1.6.6. Валидатор

Валидатор используется для проверки размеченных текстов на этапах разметки и проверки (операции 6, 11, 16).

Валидатор проверяет размеченные документы по следующим параметрам:

– синтаксическая правильность (баланс скобок и кавычек, правильная вложенность команд);

– семантическая правильность на основе DTD (допустимость команд и атрибутов);

– правильная структура информационного объекта, наличие необходимых структурных элементов и метаданных;

– наличие мишеней гипертекстовых ссылок.

Валидатор реализован как набор внешних программ, интегрированный с редактором. В частности, возможен переход из окна протокола ошибок на соответствующее место в редакторе.

1.6.7. Визуализатор

Визуализатор используется для визуальной проверки размеченных текстов на этапах разметки и проверки (операции 6, 11, 16).

Визуализатор должен правильно отображать оформление текста, отображать структуру произведения и дерево издания, отрабатывать гипертекстовые ссылки и т.д.

В качестве визуализатора используется Internet Explorer с дополнительными надстройками. Браузер интегрирован с редактором и может запускаться прямо из редактора.

1.6.8. Репозиторий объектов

Репозиторий объектов представляет собой централизованную базу данных, обеспечивающую хранение, учет, обработку и архивацию всех объектов системы на всех этапах обработки, вместе с сопутствующей мета-информацией. При этом сами объекты могут храниться во внешних файлах, но доступ к ним осуществляется только через репозиторий.

Репозиторий обеспечивает следующие возможности:

– централизованное хранение всех информационных объектов (текстов, метаданных, структур и т.д.);

– хранение и обработка древовидных структур;

– хранение различных версий объекта, автоматический контроль версий;

– контроль технологических этапов обработки объекта;

– автоматическое копирование и архивирование информации;

– возможность коллективной (распределенной) обработки информации;

– экспорт/импорт объектов в систему в различных режимах и форматах;

– поиск и выборка объектов по любым атрибутам;

– формирование проекций базы данных и выполнение с ними групповых операций (копирование, архивирование, глобальные замены);

– генерация готового электронного издания из всего массива объектов или его проекции (!).

Репозиторий является единственным местом, где фиксируется вся информация для электронного издания на всем протяжении его подготовки.

Вначале создается набросок основного дерева, где только намечены позиции для веток, изданий, произведений, и фиксируется в репозитории. Перед началом обработки любой книги для нее создается место в репозитории. В процессе обработки книга разбивается на произведения, для которых также создаются места в репозитории. Параллельно и независимо от обработки текста может создаваться метаинформация, которая попадает на заготовленные места в репозитории. Последующие модификации основного дерева не должны

изменять зафиксированные деревья изданий, так как они связаны с внешними файлами, но можно создавать альтернативные деревья, использующие те же файлы.

Обработанные файлы с текстами впервые попадают в репозитории после распознавания и вычитки (после этапа 5); все последующие версии файлов также сохраняются в репозитории. Номер версии файла должен строго соответствовать технологическому этапу обработки, номер подверсии может отражать мелкие модификации файла внутри этапа. Технологическая информация о файле включает:

– имя файла вместе с каталогом (должен отражать логическое деление материала);

– код издания (книги), частью которого он является (надо, чтобы книги соответствовали каталогам);

– номер версии и подверсии (стадии обработки);

– имя обработчика и дата окончания обработки.

Совокупность всех версий файла составляет историю его обработки.

Репозиторий должен иметь прозрачные и хорошо документированные форматы экспорта/импорта всей разнородной информации (текстов, описаний, деревьев). В частности, деревья и метаданные могут готовиться в текстовом редакторе в специальном формате, но должны прозрачно импортироваться в репозиторий. Сами тексты не импортируются, а репозиторий хранит только ссылки на файлы.

Репозиторий реализуется как надстройка над Microsoft Access или как программа на Visual Basic, использующая формат данных Access (Jet).

См. также рис.1 в конце документа.

2 СТАДИЯ ХРАНЕНИЯ, ОБРАБОТКИ И ПРЕДСТАВЛЕНИЯ ДОКУМЕНТОВ.

Системы, реализующие подготовку материалов для ЭИ согласно технологии, описанной в первой части, продуцируют массив данных, являющийся входным для программного комплекса, реализующего собственно загрузчик этого массива в хранилище данных системы (обычно РБД), а также представление этих данных для конечного пользователя в требумом ТЗ виде. Предметом рассмотрения части 2 станет именно та часть программного комплекса, которая реализует загрузку и хранения данных.

Ключевыми вопросами проектирования и реализации систем для рассматривамой предметной области (коллекции структурированных документов, или Электронные Издания, «ЭИ») являются задачи перечисления и стандартизации спецификаций на компоненты программного комплекса (ПК).

В данном разделе будут описаны модели и спецификации, использовавшиеся при реализации ЭИ «Литературное наследие А.С. Пушкина» (далее по тексту – ЭИ «Пушкин»), а также их преобразования.

2.1. Организация артефактов описания ПК

Мы используем базовое понятие описания как основной свод неформальных заключений о требуемом продукте. Его воплощениями в формализуемые формы являются:

- модели
- нотации
- спецификации
- ....

В контексте данной статьи под описанием далее будет пониматься совокупность вербально или графически изложенных тезисов, выводов и указаний, при наличии которой абстрактный разработчик имеет возможность реализовать ПК с конкретными требованиями.

Реализация описания на уровне кода является приложением, выполняющем возложененные на него в ТЗ функции и удовлетворяющим всем заявленным требованиям.

В данной статье, делающей акцент на серверной части WEB – приложения, реализующего ЭИ, основное внимание уделяется т.н. 1-му и 2-му уровню стандартного 3-слойного «thin-client web-based» приложения, другими словами, мы не рассматриваем функциональность клиента как такового и связанного с его реализацией Web сервисов: Web – сервера, сервлетов и т.д. То, что осталось от всего этого в модели данных, называется *интерфейс данных клиента* (ИДК) – часть схемы данных в РБД, которая используется приложением, реализующим внешний интерфейс, без дополнительных преобразований. Поскольку его структура возможно является определяемой отдельным набором требований, отличным от описания, задающего 1-ый и 2-ой слои (т.е. Сервера данных и Сервера приложений), модель ИДК воспринимается также как требование к описанию серверной части.

Таким образом, аргументом для описания является полный набор требований. Здесь приведены 3 важнейших:

- Требования от ТЗ
- Требования от ИДК
- Требования от исходных данных

Если бы мы рассматривали все 3 слоя ИС, то наши требования дополнились бы другими пунктами.

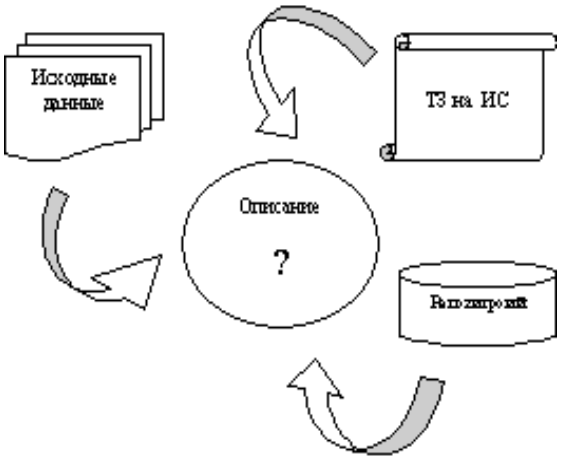


Рис.2 Требования к описанию

При переходе на формальный уровень описания мы будем считать построением Модели создание символического описания различного рода элементов системы, отражающего суть взаимодействия этих элементов, влияющего на поведение конечного (абстрактного) проектировщика, руководствующегося данным описанием. Нотацией же будет называться графический способ сопоставлению символа в модели некой фигуре, а взаимосвязи между моделями - соответственно стрелки («поток данных», «составлющая»), объединения фигур (подкомпонент) или иное. Спецификацией конкретного экземпляра модели мы будем называть перечень определений её элементов, который, собственно, и обеспечивает требуемую описанию информацию.

Основным источником для получения желаемых выводов нам дают модели, получаемые в процессе разработки системы.

2.2. Артефакты описания реализации ЭИ

Излагать описание реализации ЭИ мы будем на примере реализованной системы ЭИ «Пушкин», как типичного представителя исследуемого класса информационных систем. Из перечисленных требований существенными являются требования от ТЗ и от входных данных, поскольку структура ИДК зависела только от архитектуры клиентского интерфейса, и поэтому задача его описания выходит за рамки статьи.

Рассмотрим оставшиеся потоки требований:

2.2.1. Требования Технического Задания

Кроме аппаратных и требований к пользовательскому интерфейсу оно содержало описание основных информационных сущностей, с которыми система должна уметь работать, а также их организацию.

Таковыми были:

- Собрание сочинений
- Произведения
- Издания
- Библиография

Единственным элементом, имеющим полно-текстовое содержание, является Произведение.

Для организации таких информационных (значимых) элементов, вводились дополнительные служебные элементы типа Узел дерева, предназначенные для собрания значимых элементов в единую структуру.

Кроме этого, полный текст Произведения является иерархически структурированным при помощи элементов Элемент текста, которые хоть и не входили в список значимых, но должны быть выделены при хранении для возможности навигации по внутренности произведения.

Совокупность Элементов текста, относящихся к Произведению, образуют его полный текст.

2.2.1.1. Характеристики элементов :

- Собрание сочинений: корневой элемент, не имеющий атрибутов, кроме заглавия.
- Издание, Библиография: элементы, имеющий следующие атрибуты: Объект типа издания может включать следующие компоненты
    - идентификатор (URI),
    - имя,
    - описание,
    - формуляр - набор атрибутов (реквизитов), характеризующих публикацию как реализацию литературного произведения, например: название, автор, дата создания, дата оригинального издания, издательство т .д. Любая публикация может иметь, а может и не иметь атрибутов каждого вида. Причем допустима множественность атрибутов определенного вида для одной и той же публикации. Так, например, у издания может быть два названия, или два автора. Атрибуты используются в системе для организации поиска, а также для отображения результатов поиска в виде упорядоченного списка.
- Узел дерева: имеет заглавие и описание.
- Произведение: элемент имеет следующие компоненты:
    - идентификатор (URI),
    - имя (краткое наименование),
    - описание,
    - формуляр (то же, что и у Издания, Библиографии),
    - полный текст (опционально).
- Элемент текста: имеет URI, тип и краткое именование, а также поле данных: кусок текста с HTML разметкой.

2.2.1.2. Организация элементов

- Элементы типа Собрание сочинений, Узел дерева, Издание, Библиография могут образовывать дерево, удовлетворяющее следующим правилам:
    - Корень дерева (и только он) – Собрание сочинений.

- Связи в дереве «узел 1»-«наследник 2» имею семантическое значение «2 входит частью в 1».
- Порядок наследников первого уровня у каждой вершины важен.
- Все нетерминальные наследники у вершины типа Издание или Библиография имеют тот же тип, что у неё.
- Терминальные элементы (и только они) дерева имеют тип Произведение.

- Элементы типа Произведение, Элемент текста образуют дерево, удовлетворяющее следующим правилам:
  - Корень дерева (и только он) - Произведение.
  - Связи в дереве «узел 1»-«наследник 2» имею семантическое значение «2 входит частью в 1».
  - Порядок наследников первого уровня у каждой вершины важен.

Вводимая этим описанием система понятий (назовём её DTD2[1]), была положена в основу Информационной модели системы, составление которой состояло в получении:

- Модели информационных сущностей (ER-модель).
- Специфицирования отображения элементов DTD2 в ER-модель.

Суть шага состоит в том, что различные с точки зрения DTD2 сущности и элементы являются сходными по информационному наполнению (атрибуты) и структуре входящих частей, поэтому для их хранения возможно пользоваться одними и теми же информационными сущностями. Рассмотрим это подробнее:

Элементы типов Издание, Собрание сочинений, Библиография, Произведение, Узел дерева можно объеденить в одну информационную сущность Контейнер со следующими признаками:

- Контейнер может иметь набор упорядоченный набор входящих частей (подчинённых элементов) типа Контейнер.
- Контейнер может иметь ссылку на несколько Элементов формуляра.
- Контейнер имеет атрибуты
  - идентификатор (URI),
  - имя (краткое наименование),
  - Описание.

Элементы типа Элемент текста можно объеденить в похожую сущность Текст с тремя отличиями:

- Невозможно иметь ссылки на Элементов формуляра.
- Невозможно иметь ссылки на Описание.
- Имеет ссылку на MEMO поле содержащее фрагменты неразбираемого текста (SGML.PCDATA) и именуемое Данные.

[1] Под DTD1 в отчете по реализации ЭИ «Пушкин» понималась DTD-схема, соответствующая потоку входных данных. Из-за наличий ссылок на отчет мы сохранили обозначения.

Чтобы замкнуть это описание, необходимо специфицировать Элементы формуляра и Описания.

Исходя из желания сохранить прозрачной схему данных, все фрагменты неразбираемого текста желательно относить к одной информационной сущности Данные (но вовсе не обязательно к одной таблице в базе данных!), поэтому:

Элемент Описание

- имеет ссылку на содержащий его Контейнер.
- имеет ссылку на упорядоченный набор элементов типа Данные, образующий тело описания.

Элемент Элемент формуляра

- имеет ссылку на содержащий его Контейнер.
- имеет тип, определяющий его семантику (что это за элемент – автор, дата издания, ...). Вообще говоря, типизация элементов формуляра определяет формат МЕТА-ОПИСАНИЯ элементов ЭИ. В нашем случае это есть поля из российского библиографического формата (см. [1]), но в общем случае это может быть как произвольный набор, так и другой библиографический формат, например, Dublin Core (см. [7]).

С другой стороны, в Элементах текста (в поле данных) могут быть необъявленные в ТЗ как значимые элементы, такие как номера страниц и гиперссылки, которые, тем не менее, должны идентифицироваться на стадии хранения как адресумые в пределах Собрания сочинений, для возможности системы обращения к ним на сдадии выполнения.

Деление текста по страницам согласно требованиям к пользовательскому интерфейсу (см. [6]) требовало возможности прямой адресации к началу или концу страницы, поэтому хотя разделитель страниц являлся плавающим элементов внутри элемента Данные, указанное требование вынуждало иметь индекс этих позиций в фрагментах текста, причем с сохранением соответствия между разделителем страниц и Элементом текста и Контейнером, в которых он встретился.

Поэтому было принято следующее решение: разделители страниц считать псевдо-Элементами текста, и текстовые блоки, в которых они встречались, делить на более мелкие, чтобы внутри каждого из фрагмента Данных не встречалось разделителей. Хранить эти псевдо-Элементы текста предлагается вместе с обычными, но выделяя специальной типизацией для правильной сборки отрывков.

Гиперссылки, встречающиеся в тексте (т.е. в блоках данных элементов Данные), для использования в системе необходимо подвергать двум преобразованиям:

- изменение HREF у ссылки на URI элемента системы, соответствующего её точке назначения;
- индексация ссылок – таковая была необходима для построения т.н. Обратных ссылок, требуемых ТЗ.

Обратные ссылки представляли из себя гиперссылки системы, которых не было в исходных (грузимых) дан-

ных. Например, если в исходных данных мы имели ссылку <A id="uri1" href="uri2">***</A>, указывающую на элемент системы uri2, который мог быть и Изданием, и Произведением, и Библиографией, и Элементом Текста, то в текстовом теле (совокупность элементов Данные, ему соответствующих) элемента uri2 необходимо было добавить ссылку <A id="uri2`" href="uri1">%%%</A>, которая могла бы быть ещё и множественной, если «прямые» ссылки в количестве несокльких штук ссылались бы в один элемент данных. Поэтому вставали (и решались) следующие проблемы:

- индексация прямых ссылок – чтобы сформировать обратные ссылки необходимо было сгруппировать прямые по точкам их назначения (uri2), чтобы потом сопоставить каждой из точек назначения набор точек, которые на них ссылаются (разрешение обратных ссылок). Для хранения этого индекса был создана сущность Ассоциации. Каждый из элементов этой сущности имел как ссылку на элемент (Контейнер или Элемент Текста), который ссылается (uri1), так и разрешённую ссылку на элемент (Контейнер или Элемент Текста), на который ссылаются (uri2).
- Обратные ссылки - как их хранить. В силу того, что каждая обратная ссылка являлась именованным (через URI) элементом, содержащим текст, состоящий из нескольких гиперссылок (обратных единичных), то было решено представлять Обратные ссылки как Произведения, формируемые пост-загрузчиком, т.е. такие, которых не было во входных данных. Специфическим отличием этих Произведений, помимо этого, было то, что они не включались в дерево Контейнеров, хотя и хранились как экземпляры тех же информационных сущностей.

В ИТОГЕ: исходя из представленных заключений, была получена ER-модель. Её диаграмму (см. Рис. 3) и итоговые определения сущностей приводим:

Информационными сущностями в ней являются:

- Контейнер (CONTAINER) – основной элемент, образующий иерархическое дерево структурных элементов коллекции, находящихся вне элементов типа «ПРОИЗВЕДЕНИЕ» посредством fish-hook связи на себя через внешний ключ root_id. Он обладает возможностью иметь ссылку на элементы мета-описания и комментарии (см. ниже).
- Элемент текста (TEXTS) – то же самое что и контейнер, но для структурных элементов данных внутри элементов типа «ПРОИЗВЕДЕНИЕ». Ссылок на мета-данные и комментарии не имеет.

Надо сказать, что элементы типа «ПРОИЗВЕДЕНИЕ» выделяются тем, что структурные элементы входных данных, имеющие такую типизацию, соотвествуют базовым элементам учета при подготовке материалов (см. Раздел 1), что выразилось в формате входных данных для системы, а также обладают «самодостаточностью» в смысле ТЗ на систему, что выражается в наличии у них мета-описания (обычно – биб. карточка) и возможности быть показанными при помощи пользовательского интерфейса а) вне контекста остальных элементов; б) цельно, т.е. только вместе со всеми внутренними структурными элементами.

- Ассоциации (LINKS) – образует индекс всех гиперпереходов между структурными элементами. Помимо смены адресов перехода с уникальных имен, содержащихся во входных данных на уникальные имена в базе данных (URI), этот индекс служит для формирования ссылок, не имеющихся во входных данных: они образуются как обратные по отношению к прямым ссылкам, а значит, могут быть множественными гипер-ссылками, т.е. такими, которые имеют одну точку отправления, но много точек назначения.

- Запись формуляра (FORMULAR_RECORDS) – образуется полями мета-данных, относящихся к контейнерам отношением N-1 (каждый контейнер может иметь много записей мета-описания). Типизация полей мета-описания не противоречит действующему российскому библиографическому формату (см. [1]).

- Описание (DESCRIPTION) – содержит собранную биб-карточку в формате HTML3.2 или комментарий для контейнера. Каждый контейнер может иметь много связанных с ним элементов типа комментарий.

- Данные (PCDATA) – собственно «сплошные» куски текста, из которых состоит тело структрированного элемента. Могут входить как составные части только в элемент текст. В системе эта сущность хранила MEMO поля с HTML текстом, причем его разметка в дальнейшем интерпретировалась только как процедурная, без структурных тегов. Они же выбирались из входного потока на этапе разбора данных.

2.2.2. Формат входных данных

Данные, загружаемые в систему, имели собственную организацию и форматы, описание которых было необходимо для преобразованием их в информационные сущности системы.

Файлы, содержавщие загружаемые данные, были представлены в 2-х форматах:

- TLT – файлы (первый тип)
- HTML 3.0 – файлы. (второй тип)

Файлы первого вида были предназначены для хранения иерархии Контейнеров (т.е. связей типа «элемент»-«включённые элементы» с указанием порядка), кратких

имен каждого из элементов и ссылок на файлы второго типа, хранящие остальные данные об Контейнерах.

Формат строк в файлах 1-ого типа был следующий:

Parent_Order:Title:[HREF_TO_HTML] <CR>

Каждая строка соответствовала одному элементу типа Контейнер, и Parent_Order - глубина элемента относительно первой записи, Title - краткое наименование, [HREF_TO_HTML] - (опционально) ссылка на файл второго типа, хранящего остальную информацию об элементе.

Порядок наследников в теле отца совпадал с порядком строк в файле.

Вообще говоря, вся иерархия Контейнеров могла описываться одним экземпляром файла 1-ого типа.

Файлы 2-ого типа были предназначены для хранения:

- Полного текста произведений с процедурной разметкой HTML 3.0.
- Структурного деление полных текстов.
  - Структура 1: фрагменты текста: главы, разделы, параграфы и т.д.
  - Структура 2: деление на страницы.
- Полей мета-информации о Контейнерах.

Каждый из файлов второго типа соответствовал одному и только одному Контейнеру, а значит, одной и только одной записи в файле первого типа.

2.2.2.1. Основные переопределения

Для хранения указанной информации некоторым тегам стандартной разметки HTML 3.0 сопоставлялась оригинальная семантика, позволяющая хранить структурное деление текста и мета-информацию о элементах системы. Переопределения фиксировались только на уровне документации по входным данным и для их обработки требовали специального (ручного) кодирования программы загрузки, в чём и состояла основная задача этапа загрузки.

- Структура 1 форматировалась при помощи тега <H4> в следующем виде:
  <H4 L=Level id=”URI” title=”Name”>
  ...(тело фрагмента)…
  </H4>
  Где:
  Level – глубина вложения тега относительно содержащего его произведения,
  URI – уникальный текстовый идентификатор фрагмента,
  Name – краткое именование фрагмента.
  Иерархия элементов <H4> допускала произвольную глубину вложенности.
- Структура 2 (разделители страниц) задавалась элементами
  <SPAN class=page id=$page_number> page_number </SPAN>
- Элементы мета-описаний для Контейнеров внутри файлов 2-ого типа задавались при помощи тегов <META> в следующем виде:
  <META name=Type content="Value">,
  Где:
  Type – (типизатор записей мета-информации) определял, что хранит данная запись. Например, его значениями были AUTHOR, TITLE, DATE и т.д. Каждый из Контейнеров мог иметь произволное количество значений каждого типа – правильность их интерпритации проверялась только на этапе вывода информации на клиента. Полный список возможных типов указывался только в документации по входной информации.
  Value – текстовое значение каждого элемента мета-описания.

2.2.2.2. Преобразование от исходных данных

Под такой задачей подразумевается сопоставление формата входных данных элементам информационной модели, позволяющее специфицировать логику программы загрузчика системы. Изначальное требование к загрузчику не включало необходимость частичных загрузок\обновлений данных в системе, поэтому под работой загрузчика подразумевалась только полная загрузка данных в систему, т.е. обработка всех файлов типа 1 и 2, имеющихся на входе.

Преобразование задавалось как набор правил по сопоставлению описанным элементам входных файлов элементов схемы данных системы а также преобразование их атрибутов.

Приводим правила (основные):

- Каждой записи в файле первого типа сопоставляется Контейнер с кратким наименованием, TLT.Title; внешний ключ, указывающий контейнера отца, получался из анализа поля TLT.Parent_Order, а порядок в теле отца – из относительного порядкового номера строки в пределах одного структурного предка первого уровня.
- В виде URI контейнера использовалось TLT.HREF_TO_HTML, а для Элемента текста - TLT.HREF_TO_HTML.H4_id_value
- Каждой из записей <META> ставился в соответствие экземпляр сущности Элемент формуляра с типом META.Type и внешним ключом на Контейнер, соответствующий файлу второго типа, в котором тег <META> содержался.
- Каждому из элементов <H4> сопоставлялся Элемент текста с указанным URI, и кратким наименованием H4.title.value. Также Элемент текста получал ссылку (внешний ключ) на своего структурного отца в иерархии – это мог быть либо Элемент текста, его включающий, либо Контейнер, если <H4> имел Level=0.

В итоге, анализ требований техничекого задания на систему, анализ имеющегося на входе набора данных и построение отображения формата входных данных в

информационную модель данных позволили специфицировать программу загрузчик, который осуществляет заполнение системы данными, что и было успешно проделано в ИС «Пушкин».

ЗАКЛЮЧЕНИЕ

В статье представлен набор готовых решений для части программного комплекса, реализующего подготовку и изготовление ЭИ. Предметом дальнейшей работы является развитие этого направления с целью обеспечить единое и замкнутое моделирование всего жизненного цикла ЭИ. Первые задачи, которые необходимо решить на этом пути:

- Единая объектная модель 3-х слойной системы поддержки ЭИ (сервер-сервер приложений – сервер данных). Язык моделирования – UML.
- Стандартизация требований к пользовательскому интерфейсу ЭИ. Более четкое ограничение предметной области. Стандартизация формата входных данных. Наболее вероятный формат – Open Ebook Specification. (см. [15])
- Разработка архитектуры системы-конструктора, ориентированной на возможность реализации широкого класса ЭИ без дополнительного программирования, а лишь при помощи настроек конструктора. Предпологаемая платформа – Java.

Литература

[1] Библиографическое описание документа: ГОСТ 7.1-84.

[2] СМО ВЦ РАН, Реализация ЭИ «Литературное наследие А.С. Пушкина», Технический отчет, Июнь 2000.

[3] Единая система конструкторской документации (ЕСКД): ГОСТ 2.701-84.

[4] Rational Unified Process, http://www.rational.com/products/rup/index.jtmpl.

[5] CDM — метод разработки информационных систем фирмы Oracle, Русское издание Oracle Magazine - №2(4) 1997г.

[6] «Требования к информационной системе электронного научного издания», ТЗ на ИС «Литературное наследие А.С.Пушкина»,Москва, 1999.

[7] Dublin Core Metadata Initiative, http://www.ietf.org/rfc/rfc2413.txt

[8] Metadata Semantics Shared Across Languages: Dublin Cores in languages other than English, Thomas Baker, baker@gmd.de, National Science and Technology Development Agency, Bangkok, Thailand.

[9] Антопольский А. Б., Вигурский К. В. Проблемы создания электронных библиотек Международная выставка-ярмарка "Рынок информации-99" Москва, 28-30 апреля 1999 г.

[10] Антопольский А.Б., Вигурский К.В. "Электронный фонд "Русская литература и фольклор": основные задачи и принципы создания" // Материалы конференции "Электронные изображения и визуальные искусства" (EVA'99 Москва), М., 1999

[11] Антопольский А.Б., Вигурский К.В. Электронные библиотеки // Материалы конференции "Электронные изображения и визуальные искусства" (EVA'99 Москва), М., 1999

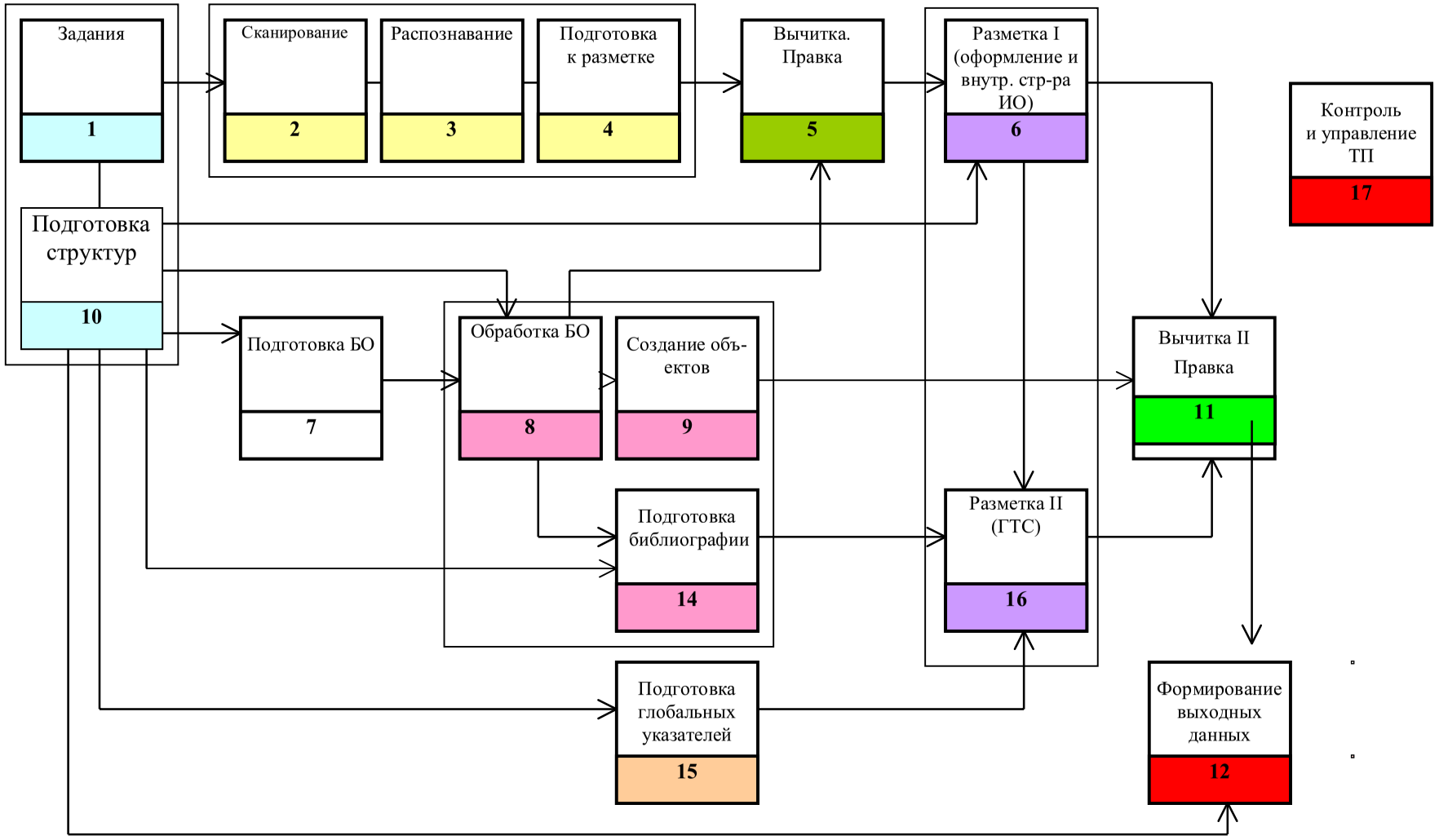
[12] Антопольский А.Б., Вигурский К.В. Проблемы электронных библиотек Доклад на 4-ом научно-исследовательском семенаре Б. Семеновкера, РГБ, май 1999г. – Библиотековедение, 1999, № 4-6. – с. 82-95

[13] Антопольский А.Б., Вигурский К.В. Электронные библиотеки. – Информационные ресурсы России 1999, №4. – с. 17-21

[14] Антопольский А.Б., Вигурский К.В., Технология подготовки электронных изданий // Материалы конференции НТИ'99. - М.,ВИНИТИ, 1999 г.

[15] Open eBook Specification, http://www.openebook.org/

Рис.2 Технология подготовки информации для ЭБ И ЭИ



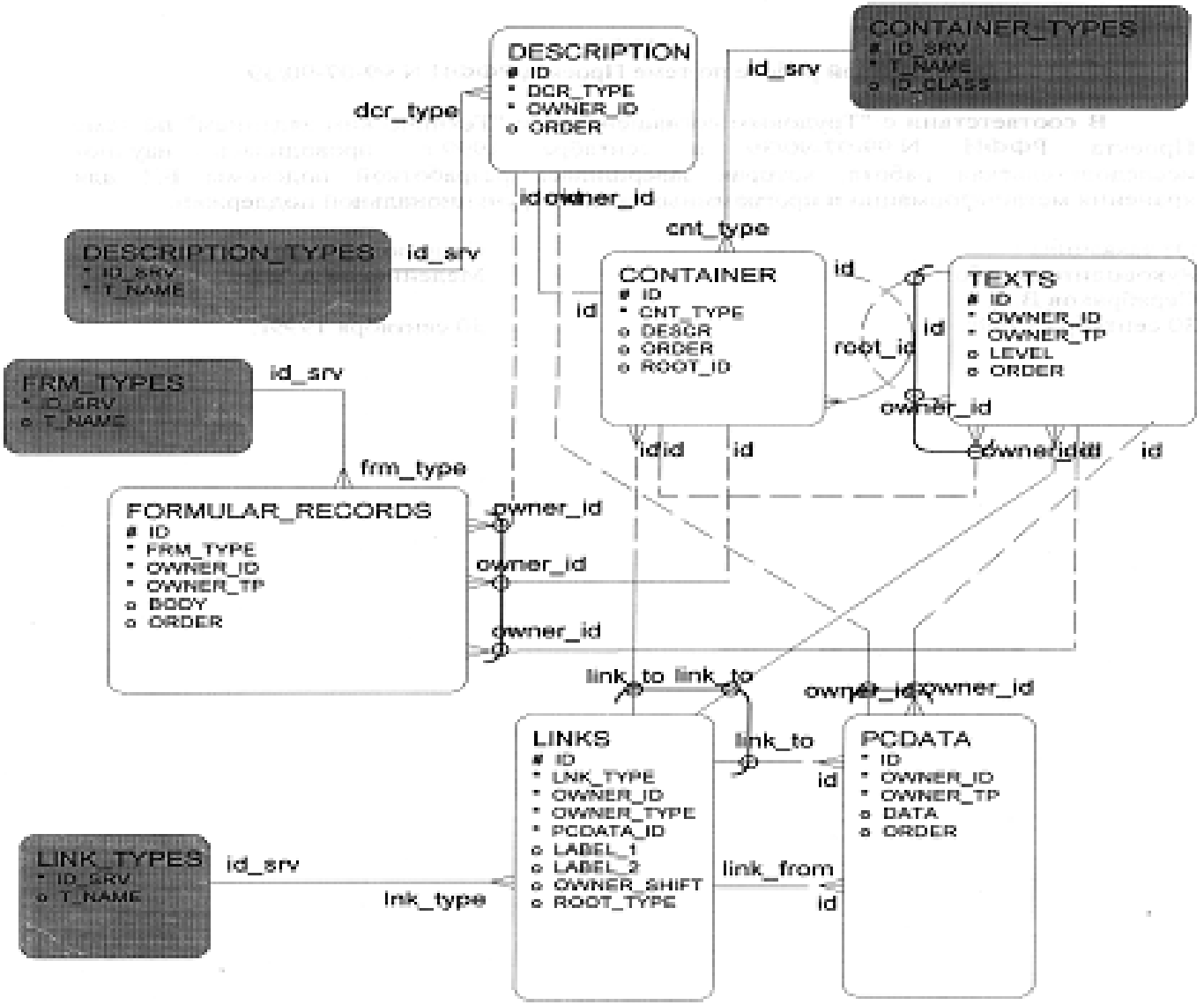


Рис 3. Модель информационных сущностей для ЭИ «Пушкин»